pt() - a2	1-2	3-4	5-6	7-8	9-10	feedback
Collaboration	The team doesn't use GitHub	The team uses version control but not all social features of GitHub	The team uses version control and the social features of GitHub to work on the project	The team worked with branches, pull requests and issues to collaborate more easily	Each team member has equal contributions and tasks are delegated between team members; they went full scrum	
Evaluation	Team members didn't review any code	Team members looked at some code but made no comments	Team members reviewed each other's code and processed issues and bugs other people assigned to them	Team members reviewed code and actively refactor code to make the application more coheren	Team members also have complex technical discussions in the comments of issues	
Application	The application doesn't work; there are errors and warnings	The application works but is incomplete; the flow of the application doesn't make sense	The team created a working dynamic prototype of a matching application	The aplication is advanced and is technically complex; it's well designed visually	The user experience is fantastic and the application is complex. The team took special care of the interface and the user	
Quality	The project is handed in documented, on time, working without technical problems, and on GitHub	The code is readable, consistent and the code, project, and process are partially documente	Code adheres to standards; docs cover the process and what the project is and does; a live version is deployed	Code quality is good and enforced; docs are more than useful and professional	Code and docs both read like great books and the project is structured logically	

You'll need a > 5.5 for each row to pass: you can't compensate between rows.
Each of this rubric’s rows is cumulative: for example, to get a 5-6 on application, you also need to have a 1-2 and 3-4.

student name	student number	lecturer	date (first chance)	grade		